Static Worksharing Strategies for Heterogeneous Computers with Unrecoverable Failures

Anne Benoit, Yves Robert, Arnold Rosenberg, Frédéric Vivien

Static Worksharing Strategies for Heterogeneous Computers with Unrecoverable Failures

Anne Benoit[1,2], Yves Robert[1,2,3], Arnold Rosenberg[4], and Frédéric Vivien[1,2,5]

[1] Ecole Normale Supérieure de Lyon, France
{Anne.Benoit,Yves.Robert,Frederic.Vivien}@ens-lyon.fr
[2] Colorado State University, Fort Collins, USA
rsnbrg@cs.umass.edu

LIP Research Report RR-2009-23

Abstract. One has a large workload that is "divisible" (its constituent work's granularity can be adjusted arbitrarily) and one has access to p remote computers that can assist in computing the workload. How can one best utilize the computers? Two features complicate this question. First, the remote computers may differ from one another in speed. Second, each remote computer is subject to interruptions of known likelihood that kill all work in progress on it. One wishes to orchestrate sharing the workload with the remote computers in a way that maximizes the expected amount of work completed.
We deal with three distinct problem instances. The simplest problem ignores communication costs, but considers a heterogeneous set of resources that may differ in speed. The other two problems account for communication costs, first with identical remote computers, and then with computers that may differ in speed. We provide exact expressions for the optimal work expectation for all three problems. For the first two problems we provide explicit, closed-form expressions; for the last (and most general) problem, we provide a recurrence for computing this optimal value.

A short version of this report appears in *HeteroPar 2009*, the International Conference on Heterogeneous Computing, jointly published with *Euro'Par 2009*, LNCS Springer Verlag.

Acknowledgments. A. Benoit, Y. Robert, and F. Vivien are with Université de Lyon, France. Y. Robert is with the Institut Universitaire de France. F. Vivien is with INRIA, France. The work of A. Benoit and Y. Robert was supported in part by the ANR StochaGrid project. The work of A. Rosenberg was supported in part by US NSF Grant CNS-0842578.

1 Introduction

This paper extends classic results from divisible load theory [10] concerning master-slave computing platforms. Our goal is to optimally distribute a given workload to p remote "slave" computers that may differ in speeds. The remote computers are connected to the "master" via a bus or network, and the master sends serially a fraction of the load to each one. The problem is to determine what fraction of the load should be sent to each remote computer, and in which order. This problem has received considerable attention in the recent past, and closed-form formulas have been derived to compute these load fractions [6, 4].

We revisit this problem in the context of remote computers that are subject to *unrecoverable failures* [5], and we aim to maximize the expected amount of total work that will be completed. For intuition: An unrecoverable failure may correspond to a hardware crash, an event more and more likely to occur with the advent of massively parallel grid platforms [1, 2]. But an unrecoverable failure may also happen with the sudden return of the user/owner in a cycle-stealing episode [3, 9, 11]. Consider the following scenario: on Friday evening, a PhD student has a large set of simulations to run. S/he has access to a set of computers from the lab, but each computer can be reclaimed at any instant by its owner. In any case, everybody will be back to work on Monday 8am. What is the student's best strategy? How much simulation data should s/he send to, and execute on, each accessible computer?

We cleave to the preceding scenario and assume that the remote computers are subject to a known risk that grows with time. More precisely: *the probability that a computer will be interrupted increases linearly with the time the computer has been available.* Other failure probability distributions could be envisioned, but the linear distribution is very natural in the absence of further information. Also, the linear risk function turns out to be tractable, in the sense that we have succeeded in deriving optimality results for this distribution. The major achievement of this paper is to provide the optimal distribution strategy to maximize the expected total amount of work done.

The paper is organized as follows. First we describe the formal framework in detail, in Section 2. Then we address three optimization problems. The simplest problem ignores communication costs, but considers a heterogeneous set of resources that may differ in speed (Section 3). The other two problems account for communication costs, first with identical remote computers (Section 4) and then with computers that may differ in speed (Section 5). We provide exact expressions for the optimal work expectation for all three problems. For the first two problems we provide explicit, closed-form expressions; for the last (and most general) problem, we have to resort to a complicated recurrence formula that provides the optimal solution in linear time. Then we provide a brief overview of related work in Section 6; in particular, we compare our approach and results with those of our previous work [5]. Finally, we give some conclusions and perspectives in Section 7.

2 Framework

We have W units of divisible work to execute on p remote computers. Each of these p computers is susceptible to unrecoverable interruptions that "kill" all work in progress (on that resource). All remote computers share the same perfectly known instantaneous probability of being interrupted, and this probability increases with the amount of time the computer has been operating (whether working or not). Within our model, all computers share the same *risk function*, i.e., the same instantaneous probability, $Pr(T)$, of having been interrupted by the end of the first T time units.

The risk function that is the focus of our study is the linear function $Pr(w) = \kappa w$. It is the most natural model in the absence of further information: the risk of interruption grows linearly with the time that the computer has been available, or equivalently with the amount of work it could have done. The density

function is then $dPr = \kappa dt$ for $t \in [0, 1/\kappa]$ and 0 otherwise, so that

$$Pr(T) = \min\left\{1, \int_0^T \kappa dt\right\} = \min\{1, \kappa T\}.$$

We suppose that all p computing remote computers obey the same probability failure distribution. For instance in the above cycle-stealing scenario, the remote computers are computers from the CS department that can be loaned during the week-end, so they have the same probability of having their owner returning. With more information about the owners, we could refine the scenario and assume different laws for, say, students and staff.

The speed of computer P_i is speed_i. The computers are interconnected by a bus or homogeneous network of bandwidth bw. Each computer will receive a single message from the master that contains all the data necessary to execute its fraction of work: this is a single-round distribution strategy with the terminology of [6]. Communications are done sequentially to each receiving computer, as in the classical divisible load model of [6]. This corresponds to a (somewhat pessimistic) *one-port* model [8], with single-threaded execution and blocking send/receive MPI primitives [12].

We introduce two important notations:

- $\mathsf{z} = \frac{\kappa}{\mathsf{bw}}$, the failure-rate per unit-load communication from the master to any computer;
- $\mathsf{x}_i = \frac{\kappa}{\mathsf{speed}_i}$, the failure-rate per unit-load computation by computer P_i.

These notations are used as follows. Suppose for example that we send a load w_1 to computer P_1, and then a load w_2 to computer P_2. The expected amount of work executed by P_1 is

$$E_1 = w_1 \left(1 - (\mathsf{z} + \mathsf{x}_1) w_1\right).$$

To understand this formula, simply observe that P_1 is communicating during the first w_1/bw time-units and computing during the next w_1/speed_1 time-units. Its risk of being interrupted linearly increases with elapsed time, regardless of whether it is communicating or computing. Similarly, we derive that the expected amount of work executed by P_2 is

$$E_2 = w_2 \left(1 - \mathsf{z}(w_1 + w_2) - \mathsf{x}_2 w_2\right).$$

Indeed, P_2 has started computing only after both communications from the master to P_1 and P_2 have completed, which takes $(w_1 + w_2)/\mathsf{bw}$ time-steps; then it computes during w_2/speed_2 additional time-steps. Again, its risk of being interrupted linearly increases with elapsed time, regardless of whether it is waiting (while the master communicates to P_1), or communicating, or computing. If we had only these two remote computers ($p = 2$), we would aim at maximizing $E_1 + E_2$, the expected total amount of work done.

Finally, we make a technical assumption and assume that the total load is small enough so that we distribute it entirely to the p computers. Indeed, if the total load is too large, all computers will fail with probability one before completing their chunk. In the following we assume that the p chunks received by the computers form a partition of the original load, and that there is a non zero probability that the last computer does not fail before or during its computation. A sufficient condition for this latter condition to hold is

$$W \leq \frac{1}{\mathsf{z} + \mathsf{x}_{\max}},$$

where $\mathsf{x}_{\max} = \frac{\kappa}{\min_{1 \leq i \leq p} \mathsf{speed}_i}$ is the failure-rate per unit-load computation of the slowest computer. To see this, simply note that the last computer, say P_i, can always start computing at time-step Y/bw, where $Y \leq W$ is the total load sent to all preceding computers: introducing idle times in the communication cannot improve the solution, as the failure risk grows with time. Then P_i needs V/speed_i time-steps to execute its own chunk of size V, where $Y + V \leq W$, hence the claim.

We can now formally state the optimization problem:

Definition 1. *We let* DISTRIB(p) *denote the problem to compute* $\mathcal{E}^{opt}(W,p)$*, the optimal value of the expected total amount of work done when distributing the entire workload* $W \leq \frac{1}{\mathsf{z}+\mathsf{x}_{\max}}$ *to the* p *remote computers.*

In the following, we first deal with the case where $\mathsf{z} = 0$, i.e., when communication costs can be neglected. Then we tackle the case with communication costs and identical remote computers ($\mathsf{x}_i = \mathsf{x}$ for $1 \leq i \leq p$) before moving to the general case with communication costs and different-speed remote computers.

3 Heterogeneous remote computers, no communication costs

In this section we deal with the DISTRIB problem when we have p different-speed remote computers but do not pay any communication cost. This models the case where computations are prevailing in the application. We need to introduce symmetric functions to state the result:

Definition 2. *Given* $n \geq 1$*, for* $0 \leq i \leq n$*,* $\sigma_i^{(n)}$ *denotes the* i*-th symmetric function of* $\mathsf{x}_1, \mathsf{x}_2, \ldots, \mathsf{x}_n$*:*

$$\sigma_i^{(n)} = \sum_{1 \leq j_1 < j_2 < \cdots < j_i \leq n} \prod_{k=1}^{i} \mathsf{x}_{j_k}.$$

By convention $\sigma_0^{(n)} = 1$*.*

For instance with $n = 3$, $\sigma_1^{(3)} = \mathsf{x}_1 + \mathsf{x}_2 + \mathsf{x}_3$, $\sigma_2^{(3)} = \mathsf{x}_1\mathsf{x}_2 + \mathsf{x}_1\mathsf{x}_3 + \mathsf{x}_2\mathsf{x}_3$ and $\sigma_3^{(3)} = \mathsf{x}_1\mathsf{x}_2\mathsf{x}_3$.

Theorem 1. *When* $\mathsf{z} = 0$ *the optimal solution to* DISTRIB(p) *is obtained when the chunk sent to* P_i *is of size* $\frac{\prod_{k \neq i} \mathsf{x}_k}{\sigma_{p-1}^{(p)}} W$*, leading to*

$$\mathcal{E}^{opt}(W,p) = W - \frac{\sigma_p^{(p)}}{\sigma_{p-1}^{(p)}} W^2.$$

Proof. Let $\alpha_{i,p} = \frac{\prod_{k \neq i} \mathsf{x}_k}{\sigma_{p-1}^{(p)}} W$ and $\mathsf{f}_p = \frac{\sigma_p^{(p)}}{\sigma_{p-1}^{(p)}}$. We show the result by induction. Note that it holds for $p = 1$, because $\alpha_{1,1} = 1$ and $\mathsf{f}_1 = \mathsf{x}_1$.

To help the reader follow the derivation, we prove the result for $p = 2$ before dealing with the general case. Assume that the size of the chunk sent to P_1 is Y. The size of the chunk sent to P_2 is thus $W - Y$. Both chunks are sent in parallel, as no cost is assessed for communications. The expected amount of work done is

$$E(Y) = Y\left(1 - \mathsf{x}_1 Y\right) + (W - Y)\left(1 - \mathsf{x}_2 (W - Y)\right).$$

We rewrite

$$E(Y) = W - \mathsf{x}_2 W^2 - (\mathsf{x}_1 + \mathsf{x}_2) Y^2 + 2\mathsf{x}_2 W Y.$$

The optimal value is $Y^{(\text{opt})} = \frac{\mathsf{x}_2}{\mathsf{x}_1 + \mathsf{x}_2} W = \alpha_{1,2} W$ as desired (and $W - Y^{(\text{opt})} = \frac{\mathsf{x}_1}{\mathsf{x}_1 + \mathsf{x}_2} W = \alpha_{2,2} W$). Reporting the value of $Y^{(\text{opt})}$ into the expression of $E(Y)$, we derive that

$$\mathcal{E}^{\text{opt}}(W,2) = E(Y^{(\text{opt})}) = W - \mathsf{f}_2 W^2,$$

where

$$\mathsf{f}_2 = \mathsf{x}_2 - \frac{\mathsf{x}_2^2}{\mathsf{x}_1 + \mathsf{x}_2} = \frac{\mathsf{x}_1 \mathsf{x}_2}{\mathsf{x}_1 + \mathsf{x}_2} = \frac{\sigma_2^{(2)}}{\sigma_1^{(2)}}.$$

This proves the claim for $p = 2$.

Assume now that the results holds up to n computers. Consider the case of $n+1$ computers, and assume that the size of the chunk sent to P_{n+1} is $W - Y$. By induction, the optimal expected amount of work done by the first n computers is $Eopt(Y, n) = Y\,(1 - \mathsf{f}_n Y)$, and this is achieved by sending a chunk of size $\alpha_{i,n} Y$ to P_i for $1 \leq i \leq n$. The expected amount of work done is then

$$E(Y) = Y\,(1 - \mathsf{f}_n Y) + (W - Y)\,(1 - \mathsf{x}_{n+1}(W - Y))\,.$$

We proceed just as above. The optimal value is $Y^{(\text{opt})} = \frac{\mathsf{x}_{n+1}}{\mathsf{f}_n + \mathsf{x}_{n+1}} W$ and we derive $\mathcal{E}^{\text{opt}}(W, n+1) = E(Y^{(\text{opt})}) = W - \mathsf{f}_{n+1} W^2$ where

$$\mathsf{f}_{n+1} = \mathsf{x}_{n+1} - \frac{\mathsf{x}_{n+1}^2}{\mathsf{f}_n + \mathsf{x}_{n+1}}.$$

We recognize that

$$\sigma_n^{(n)} + \mathsf{x}_{n+1}\sigma_{n-1}^{(n)} = \sigma_n^{(n+1)}$$

so that $\mathsf{f}_n + \mathsf{x}_{n+1} = \frac{\sigma_n^{(n+1)}}{\sigma_{n-1}^{(n)}}$ and

$$\mathsf{f}_{n+1} = \mathsf{x}_{n+1} - \frac{\mathsf{x}_{n+1}^2 \sigma_{n-1}^{(n)}}{\sigma_n^{(n+1)}} = \frac{\mathsf{x}_{n+1}\left(\sigma_n^{(n+1)} - \mathsf{x}_{n+1}\sigma_{n-1}^{(n)}\right)}{\sigma_n^{(n+1)}} = \frac{\mathsf{x}_{n+1}\sigma_n^{(n)}}{\sigma_n^{(n+1)}} = \frac{\sigma_{n+1}^{(n+1)}}{\sigma_n^{(n+1)}}$$

as desired.

Also, $Y^{(\text{opt})} = \frac{\mathsf{x}_{n+1}}{\mathsf{f}_n + \mathsf{x}_{n+1}} W = \frac{\mathsf{x}_{n+1}\sigma_{n-1}^{(n)}}{\sigma_n^{(n+1)}} W$. By induction, for $1 \leq i \leq n$, we get

$$\alpha_{i,n+1} = \alpha_{i,n} \frac{\mathsf{x}_{n+1}\sigma_{n-1}^{(n)}}{\sigma_n^{(n+1)}} = \frac{\mathsf{x}_{n+1}\sigma_{n-1}^{(n)} \prod_{1 \leq k \leq n, k \neq i} \mathsf{x}_k}{\sigma_{n-1}^{(n)}\sigma_n^{(n+1)}} = \frac{\mathsf{x}_{n+1} \prod_{1 \leq k \leq n, k \neq i} \mathsf{x}_k}{\sigma_n^{(n+1)}} = \frac{\prod_{1 \leq k \leq n+1, k \neq i} \mathsf{x}_k}{\sigma_n^{(n+1)}}$$

as desired. It remains to check the value of

$$\alpha_{n+1,n+1} = 1 - \frac{\mathsf{x}_{n+1}\sigma_{n-1}^{(n)}}{\sigma_n^{(n+1)}} = \frac{\sigma_n^{(n+1)} - \mathsf{x}_{n+1}\sigma_{n-1}^{(n)}}{\sigma_n^{(n+1)}} = \frac{\prod_{1 \leq k \leq n} \mathsf{x}_k}{\sigma_n^{(n+1)}}$$

which concludes the proof. □

We see that the optimal solution is *symmetric*: the contribution of each computer is a (somewhat complicated but) symmetric function of all computer speeds.

4 Homogeneous remote computers, with communication costs

We now move to the case with communication costs. Before dealing with the general case of heterogeneous remote computers, which turns out to be difficult, we address the problem with identical remote computers:

Theorem 2. *When $\mathsf{x}_i = \mathsf{x}$ (identical speeds), the optimal solution to* DISTRIB(p) *is obtained with same size chunks (hence of size $\frac{W}{p}$). The optimal expected total amount of work done is*

$$\mathcal{E}^{opt}(W, p) = W - \frac{(p+1)\mathsf{z} + 2\mathsf{x}}{2p} W^2.$$

Proof. The proof is similar to that of Theorem 1. Let $\mathsf{f}_p = \frac{(p+1)\mathsf{z}+2\mathsf{x}}{2p}$. We show the result by induction. Note that it holds for $p = 1$, because $\mathsf{f}_1 = \mathsf{z} + \mathsf{x}$.

Assume that the result holds for up to n computers. Consider the case of $n+1$ computers, and assume that the size of the chunk sent to P_{n+1} is $W - Y$. By induction, the optimal expected amount of work done by the fist n computers is $Eopt(Y, n) = Y\,(1 - \mathsf{f}_n Y)$, and this is achieved by sending a chunk of size $\frac{Y}{n}$ to each P_i, $1 \leq i \leq n$. The expected amount of work done is then

$$E(Y) = Y\,(1 - \mathsf{f}_n Y) + (W - Y)\,(1 - \mathsf{z}W - \mathsf{x}(W - Y))\,.$$

To understand the value of the contribution of P_{n+1} to $E(Y)$, simply note that it has to wait for the whole workload to be distributed (accounted for by the term $\mathsf{z}W$) before it can start computing its own chunk (accounted for by the term $\mathsf{x}(W - Y)$). We rewrite $E(Y)$ as

$$E(Y) = W - (\mathsf{z} + \mathsf{x})W^2 - (\mathsf{f}_n + \mathsf{x})Y^2 + (\mathsf{z} + 2\mathsf{x})WY.$$

The optimal value is $Y^{(\mathrm{opt})} = \frac{\mathsf{z}+2\mathsf{x}}{2(\mathsf{f}_n+\mathsf{x})}W$ and we derive that $\mathcal{E}^{\mathrm{opt}}(W, n+1) = E(Y^{(\mathrm{opt})}) = W - \mathsf{f}_{n+1}W^2$, where

$$\mathsf{f}_{n+1} = \mathsf{z} + \mathsf{x} - \frac{(\mathsf{z} + 2\mathsf{x})^2}{4(\mathsf{f}_n + \mathsf{x})}.$$

Using the induction hypothesis, we get $\mathsf{f}_n + \mathsf{x} = \frac{(n+1)\mathsf{z}+2\mathsf{x}}{2n} + \mathsf{x} = \frac{(n+1)(\mathsf{z}+2\mathsf{x})}{2n}$, so that

$$\mathsf{f}_{n+1} = \mathsf{z} + \mathsf{x} - \frac{n(\mathsf{z} + 2\mathsf{x})}{2(n+1)} = \frac{(n+2)\mathsf{z} + 2\mathsf{x}}{2(n+1)}$$

as expected. We also obtain $Y^{(\mathrm{opt})} = \frac{n}{n+1}W$, so that each P_i (with $i \leq n$) receives a chunk of size $\frac{Y^{(\mathrm{opt})}}{n} = \frac{W}{n+1}$. We deduce that P_{n+1} receives a chunk of that same size (or we can directly check that $W - Y^{(\mathrm{opt})} = \frac{W}{n+1}$). This concludes the proof. □

Interestingly, the optimal solution involves to send same-size chunks to all computers. In contrast, in the classical divisible load setting, when we aim at minimizing the total time needed to execute a certain amount of load, all computers terminate their computations simultaneously [6], so that the first computers served by the master receive longer chunks.

5 Heterogeneous remote computers, with communication costs

We are now ready for the general case, with communication costs and different-speed computers. We need a few notations before stating the main result of this paper:

Definition 3. *We define the following sequence:* $\lambda_0 = \lambda_1 = 4$ *and for* $n \geq 2$, $\lambda_n = \lambda_{n-1} - \frac{1}{4}\lambda_{n-2}$. *For convenience we also let* $\lambda_{-1} = 0$.

Note that we can compute that $\lambda_n = \frac{4(1+n)}{2^n}$ for all $n \geq 0$. The sequence λ is used to characterize the optimal solution:

Theorem 3. *In the general case, the optimal solution to* DISTRIB(p) *does not depend upon the ordering of the communications from the master. When using the ordering* $P_1, P_2, \ldots, P_p$, *the optimal solution is obtained when the chunk sent to* P_i *is of size* $\alpha_{i,p}W$, *leading to the optimal expected amount of work done*

$$\mathcal{E}^{opt}(W, p) = W - \mathsf{f}_p W^2,$$

where

– $\mathsf{f}_p = \dfrac{\sum_{i=0}^{p} \lambda_i \sigma_{p-i}^{(p)} \mathsf{z}^i}{\sum_{i=0}^{p-1} \lambda_i \sigma_{p-i-1}^{(p)} \mathsf{z}^i}$ *for* $p \geq 1$;
– $\alpha_{p,p} = \dfrac{2\mathsf{f}_{p-1} - \mathsf{z}}{2(\mathsf{f}_{p-1} + \mathsf{x}_p)}$ *for* $p \geq 2$ *and* $\alpha_{1,1} = 1$;
– $\alpha_{i,p} = \dfrac{\mathsf{z} + 2\mathsf{x}_{i-1}}{2(\mathsf{f}_{i-1} + \mathsf{x}_i)}(1 - \alpha_{i+1,p})$ *for* $p - 1 \geq i \geq 2$;
– $\alpha_{1,p} = 1 - \alpha_{2,p}$ *for* $p \geq 2$.

Proof. The proof is similar to that of Theorems 1 and 2, but it is more involved. We show the result by induction. Note that it holds for $p = 1$, because $\mathsf{f}_1 = \frac{\lambda_0 \mathsf{x}_1 + \lambda_1 \mathsf{z}}{\lambda_0} = \mathsf{z} + \mathsf{x}_1$.

To give intuition for the result, in particular why the ordering of the communications is not important, consider the case with two computers, P_1 and P_2, and assume that they are served in this order (first P_1, then P_2). If we send a chunk of size Y to P_1 and one of size $W - Y$ to P_2 we derive that the expectation of the amount of work done is

$$E(Y) = Y\left(1 - \mathsf{f}_1 Y\right) + (W - Y)\left(1 - (\mathsf{z}W + \mathsf{x}_2(W - Y)\right).$$

As before, to understand this equation, we note that P_2 is waiting for the first chunk to be sent to P_1; then it is receiving its own chunk; both steps account for the term $\mathsf{z}W$ in the right hand side. Finally, P_2 computes its chunk, whence the term $\mathsf{x}_2(W - Y)$. We rewrite

$$E(Y) = W - (\mathsf{z} + \mathsf{x}_2)W^2 - (\mathsf{f}_1 + \mathsf{x}_2)Y^2 + (\mathsf{z} + 2\mathsf{x}_2)WY.$$

The optimal value is $Y^{(\mathrm{opt})} = \frac{\mathsf{z} + 2\mathsf{x}_2}{2(\mathsf{f}_1 + \mathsf{x}_2)} W = \alpha_{2,2} W$ and we derive that

$$\mathcal{E}^{\mathrm{opt}}(W, 2) = W - \mathsf{f}_2 W^2$$

where

$$\mathsf{f}_2 = \mathsf{z} + \mathsf{x}_2 - \frac{(\mathsf{z} + 2\mathsf{x}_2)^2}{4(\mathsf{f}_1 + \mathsf{x}_2)}$$

which after some easy manipulation becomes

$$\mathsf{f}_2 = \frac{4\mathsf{x}_1\mathsf{x}_2 + 4(\mathsf{x}_1 + \mathsf{x}_2)\mathsf{z} + 3\mathsf{z}^2}{4(\mathsf{x}_1 + \mathsf{x}_2 + \mathsf{z})},$$

as desired. We see that the formula is symmetric in x_1 and x_2, thereby showing that the ordering of the communications has no significance.

Assume now that the result is true up to n computers, and consider $n + 1$ computers that are served in the order $P_1, \ldots, P_{n+1}$. Suppose that we send a chunk of size $W - Y$ to P_{n+1}. We know by induction that the best way to distribute the remaining Y units of work to the first n computers is independent of the ordering, and that the optimal expectation $\mathcal{E}^{\mathrm{opt}}(W, n)$ is given by $\mathcal{E}^{\mathrm{opt}}(W, n) = W - \mathsf{f}_n W^2$.

The total expectation $E(Y)$ for the $n + 1$ computers is obtained as previously:

$$E(Y) = W - (\mathsf{z} + \mathsf{x}_{n+1})W^2 - (\mathsf{f}_n + \mathsf{x}_{n+1})Y^2 + (\mathsf{z} + 2\mathsf{x}_{n+1})WY.$$

The optimal value is $Y^{(\mathrm{opt})} = \frac{\mathsf{z} + 2\mathsf{x}_{n+1}}{2(\mathsf{f}_n + \mathsf{x}_{n+1})} W$ and we derive that

$$\mathsf{f}_{n+1} = \mathsf{z} + \mathsf{x}_{n+1} - \frac{(\mathsf{z} + 2\mathsf{x}_{n+1})^2}{4(\mathsf{f}_n + \mathsf{x}_{n+1})}.$$

We know by induction that $\mathsf{f}_n = \frac{\mathsf{a}_n}{\mathsf{b}_n}$, where $\mathsf{a}_n = \sum_{i=0}^{n} \lambda_i \sigma_{n-i}^{(n)} \mathsf{z}^i$ and $\mathsf{b}_n = \sum_{i=0}^{n-1} \lambda_i \sigma_{n-i-1}^{(n)} \mathsf{z}^i$. We have $\mathsf{f}_n + \mathsf{x}_{n+1} = \frac{\mathsf{a}_n + \mathsf{x}_{n+1}\mathsf{b}_n}{\mathsf{b}_n}$ and

$$\mathsf{a}_n + \mathsf{x}_{n+1}\mathsf{b}_n = \sum_{i=0}^{n-1} \lambda_i \left(\sigma_{n-i}^{(n)} + \mathsf{x}_{n+1} \sigma_{n-i-1}^{(n)} \right) \mathsf{z}^i + \lambda_n \mathsf{z}^n.$$

But we recognize that for $0 \le i \le n-1$, we have

$$\sigma_{n-i}^{(n)} + \mathsf{x}_{n+1} \sigma_{n-i-1}^{(n)} = \sigma_{n-i}^{(n+1)}. \tag{1}$$

We also have $\sigma_0^{(n)} = \sigma_0^{(n+1)} = 1$, so that

$$\mathsf{b}_{n+1} = \mathsf{a}_n + \mathsf{x}_{n+1}\mathsf{b}_n = \sum_{i=0}^{n} \lambda_i \sigma_{n-i}^{(n+1)} \mathsf{z}^i.$$

Now we report this result into the expression of f_{n+1} and we obtain $\mathsf{f}_{n+1} = \frac{\mathsf{a}_{n+1}}{\mathsf{b}_{n+1}}$, where

$$\begin{aligned}
\mathsf{a}_{n+1} &= \mathsf{b}_{n+1}(\mathsf{z} + \mathsf{x}_{n+1}) - \frac{1}{4}(\mathsf{z} + 2\mathsf{x}_{n+1})^2 \mathsf{b}_n \\
&= \left(\sum_{i=0}^{n} \lambda_i \sigma_{n-i}^{(n+1)} \mathsf{z}^i \right) (\mathsf{z} + \mathsf{x}_{n+1}) - (\frac{\mathsf{z}^2}{4} + \mathsf{x}_{n+1}\mathsf{z} + \mathsf{x}_{n+1}^2) \left(\sum_{i=0}^{n-1} \lambda_i \sigma_{n-i-1}^{(n)} \mathsf{z}^i \right) \\
&= \mathsf{z}^{n+1}(\lambda_n - \frac{\lambda_{n-1}}{4}) + \sum_{i=1}^{n} \mathsf{z}^i (A_i + B_i - C_i - D_i - E_i) + \lambda_0(\sigma_n^{(n+1)} \mathsf{x}_{n+1} - \sigma_{n-1}^{(n)} \mathsf{x}_{n+1}^2)
\end{aligned}$$

where we have

$$\begin{array}{lll}
A_i = \lambda_i \sigma_{n-i}^{(n+1)} \mathsf{x}_{n+1} & \text{for } 1 \le i \le n & \\
B_i = \lambda_{i-1} \sigma_{n-i+1}^{(n+1)} & \text{for } 1 \le i \le n & \\
C_i = \frac{\lambda_{i-2}}{4} \sigma_{n-i+1}^{(n)} & \text{for } 2 \le i \le n, & \text{and } C_1 = 0 \\
D_i = \lambda_{i-1} \sigma_{n-i}^{(n)} \mathsf{x}_{n+1} & \text{for } 1 \le i \le n & \\
E_i = \lambda_i \sigma_{n-i-1}^{(n)} \mathsf{x}_{n+1}^2 & \text{for } 1 \le i \le n-1, & \text{and } E_n = 0.
\end{array}$$

Next we use Equation (1) to derive $A_i - E_i = \lambda_i \sigma_{n-i}^{(n)} \mathsf{x}_{n+1}$, $B_i - D_i = \lambda_{i-1} \sigma_{n-i+1}^{(n)}$, then $B_i - D_i - C_i = \left(\lambda_{i-1} - \frac{\lambda_{i-2}}{4} \right) \sigma_{n-i+1}^{(n)} = \lambda_i \sigma_{n-i+1}^{(n)}$, and finally $A_i + B_i - C_i - D_i - E_i = \lambda_i \sigma_{n-i+1}^{(n+1)}$. As for the first and last terms, we get $\lambda_n - \frac{\lambda_{n-1}}{4} = \lambda_{n+1} = \lambda_{n+1} \sigma_0^{(n+1)}$ and $\lambda_0(\sigma_n^{(n+1)} \mathsf{x}_{n+1} - \sigma_{n-1}^{(n)} \mathsf{x}_{n+1}^2) = \lambda_0 \sigma_n^{(n)} \mathsf{x}_{n+1} = \lambda_0 \sigma_{n+1}^{(n+1)}$. Altogether, we do obtain

$$\mathsf{a}_{n+1} = \sum_{i=0}^{n+1} \lambda_i \sigma_{n+1-i}^{(n+1)} \mathsf{z}^i,$$

which establishes the inductive formula. Also, because the formula is symmetric, we see that the ordering of the communications has no impact. This concludes the proof of the theorem for the value of $\mathcal{E}^{\mathrm{opt}}(W, p)$.

As for the size of the chunks, we have found that $Y^{(\mathrm{opt})} = \frac{\mathsf{z} + 2\mathsf{x}_{n+1}}{2(\mathsf{f}_n + \mathsf{x}_{n+1})} W$, hence with p computers,

$$\alpha_{p,p} = 1 - \frac{\mathsf{z} + 2\mathsf{x}_p}{2(\mathsf{f}_{p-1} + \mathsf{x}_p)} = \frac{2\mathsf{f}_{p-1} - \mathsf{z}}{2(\mathsf{f}_{p-1} + \mathsf{x}_p)}$$

as desired. We proceed by induction to determine the value of $\alpha_{i,p}$ for $i = p-1$ down to $i = 2$, and then $i = 1$. With $p = 2$, we check that $\alpha_{2,2} = \frac{\mathsf{z} + 2\mathsf{x}_1}{2(\mathsf{z} + \mathsf{x}_1 + \mathsf{x}_2)}$ (remember that $\mathsf{f}_1 = \mathsf{z} + \mathsf{x}_1$) and then $\alpha_{1,2} = \frac{\mathsf{z} + 2\mathsf{x}_2}{2(\mathsf{z} + \mathsf{x}_1 + \mathsf{x}_2)}$. □

The interested reader can check that we retrieve the values of f_p and $\alpha_{i,p}$ given in Theorem 1 when $\mathsf{z} = 0$, and those given in Theorem 2 when $\mathsf{x}_i = \mathsf{x}$.

6 Related work

The divisible-load model is a reasonable abstraction of an application made up of a large number of identical, fine-grained parallel computations. Such applications are found in many scientific areas, and we refer the reader to the survey paper [10] and the journal special issue [7] for detailed examples. Also, the divisible-load approach has been applied successfully to a variety of computing platforms, such as bus-shaped, star-shaped, and even tree-shaped platforms. Despite the extensive literature on the divisible-load model, to the best of our knowledge, the current study is the first to consider the divisible-load problem on master-slave platforms whose computers are subject to unrecoverable failures/interruptions.

Our earlier work [5], and its predecessors [3, 9, 11], also consider computers with unrecoverable failures/interruptions, but with major differences in the models. In this paper, we allow *heterogeneous* computers and communication costs, while [5] focuses only on identical computers without communication costs. To "compensate" for the additional complexity in the model we study here, we have restricted ourselves in this paper to scenarios where the entire workload is distributed to the remote computers, a strategy that is often suboptimal, even when scheduling a single remote computer [5]. Furthermore, we have not considered here the possible benefits of replicating the execution of some work units on several remote computers, a key tool for enhancing expected work production in [5]. Obviously, it would be highly desirable to combine the sophisticated platforms of the current study with the sophisticated algorithmics of [5]. We hope to do so in future work, in order to deal with the most general master-slave problem instances—instances that allow heterogeneous computing resources and communication costs, that do not insist that all work be distributed, and that give the scheduler the option of replicating work on multiple remote computers.

7 Conclusion

In this paper we have revisited the well-known master-slave paradigm for divisible-load applications, adding the hypothesis that the computers are subject to unrecoverable failures/interruptions. In this novel context, the natural objective of a schedule is to maximize the expected amount of work that gets completed. We have succeeded in providing either closed-form formulas or linear recurrences to characterize optimal solutions, thereby providing a nice counterpart to existing results in the classical context of makespan minimization. In particular, our demonstration that the ordering of communications has no impact on the optimal solution is a very interesting (and somewhat unexpected) result, as it shows that the scheduling problem has polynomial complexity: there is no need to explore the combinatorial space of all possible orderings.

As discussed in Section 6, we have adopted certain simplifications to the general problem we ultimately aspire to. We have insisted on distributing the entire workload to the remote computers, without replication of work. Our not allowing work replication is particularly unfortunate when contemplating environments that have access to abundant computing resources. This, then, is the first projected avenue for extending the current work. Several other extensions of this work would be desirable also, for instance: (*i*) including a start-up overhead-cost each time a computer executes a piece of work (e.g., to account for the cost of initiating a communication or a checkpointing); (*ii*) studying computers that obey not only linear, but also different risk functions (e.g., when several user categories have different probabilities of returning to reclaim their computers); (*iii*) studying risk functions that are no longer linear (e.g., standard exponential or, importantly, heavy-tailed distributions); and (*iv*) analyzing multi-round strategies, wherein each remote computer receives its share of work in several rounds. Altogether, there are many challenging algorithmic problems to address!

References

[1] J. Abawajy. Fault-tolerant scheduling policy for grid computing systems. In *International Parallel and Distributed Processing Symposium IPDPS'2004*. IEEE Computer Society Press, 2004.

[2] S. Albers and G. Schmidt. Scheduling with unexpected machine breakdowns. *Discrete Applied Mathematics*, 110(2-3):85–99, 2001.

[3] B. Awerbuch, Y. Azar, A. Fiat, and F. T. Leighton. Making commitments in the face of uncertainty: How to pick a winner almost every time. In *28th ACM SToC*, pages 519–530, 1996.

[4] O. Beaumont, H. Casanova, A. Legrand, Y. Robert, and Y. Yang. Scheduling divisible loads on star and tree networks: results and open problems. *IEEE Trans. Parallel Distributed Systems*, 16(3):207–218, 2005.

[5] A. Benoit, Y. Robert, A. Rosenberg, and F. Vivien. Static strategies for worksharing with unrecoverable interruptions. In *IPDPS'2009, the 23rd IEEE International Parallel and Distributed Processing Symposium*. IEEE Computer Society Press, 2009.

[6] V. Bharadwaj, D. Ghose, V. Mani, and T. Robertazzi. *Scheduling Divisible Loads in Parallel and Distributed Systems*. IEEE Computer Society Press, 1996.

[7] V. Bharadwaj, D. Ghose, and T. Robertazzi. Divisible load theory: a new paradigm for load scheduling in distributed systems. *Cluster Computing*, 6(1):7–17, 2003.

[8] P. Bhat, C. Raghavendra, and V. Prasanna. Efficient collective communication in distributed heterogeneous systems. *Journal of Parallel and Distributed Computing*, 63:251–263, 2003.

[9] S. Bhatt, F. Chung, F. Leighton, and A. Rosenberg. On optimal strategies for cycle-stealing in networks of workstations. *IEEE Trans. Computers*, 46(5):545–557, 1997.

[10] T. Robertazzi. Ten reasons to use divisible load theory. *IEEE Computer*, 36(5):63–68, 2003.

[11] A. L. Rosenberg. Optimal schedules for cycle-stealing in a network of workstations with a bag-of-tasks workload. *IEEE Trans. Parallel Distrib. Syst.*, 13(2):179–191, 2002.

[12] M. Snir, S. W. Otto, S. Huss-Lederman, D. W. Walker, and J. Dongarra. *MPI the complete reference*. The MIT Press, 1996.